



Express Mail Label No. _____

Dated: _____

Docket No.: 04970/000N023-US0
(PATENT)

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Patent Application of:
Akishige Yamamoto, et al.

Application No.: 10/628,133

Confirmation No.: 4516

Filed: July 25, 2003

Art Unit: N/A

For: FUNCTION EXECUTION METHOD,
FUNCTION EXECUTION APPARATUS,
COMPUTER PROGRAM AND RECORDED
MEDIUM

Examiner: Not Yet Assigned

CLAIM FOR PRIORITY AND SUBMISSION OF DOCUMENTS

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Dear Sir:

Applicant hereby claims priority under 35 U.S.C. 119 based on the following prior foreign application filed in the following foreign country on the date indicated:

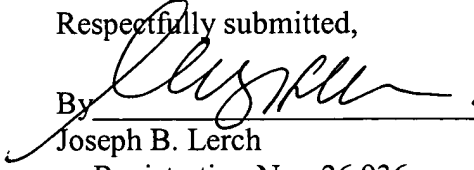
<u>Country</u>	<u>Application No.</u>	<u>Date</u>
Japan	2001-088850	March 26, 2001

In support of this claim, a certified copy of the said original foreign application is filed herewith.

Dated: December 12, 2003

Respectfully submitted,

By


Joseph B. Lerch

MARIE GILFILLAN
44085

Registration No.: 26,936

DARBY & DARBY P.C.

P.O. Box 5257

New York, New York 10150-5257

(212) 527-7700

(212) 753-6237 (Fax)

Attorneys/Agents For Applicant



日 本 国 特 許 庁
JAPAN PATENT OFFICE

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office.

出 願 年 月 日 2 0 0 1 年 3 月 2 6 日
Date of Application:

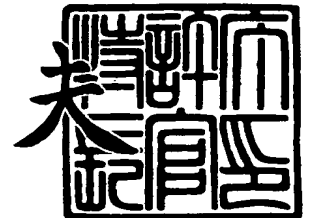
出 願 番 号 特 願 2 0 0 1 - 0 8 8 8 5 0
Application Number:
[ST. 10/C]: [J P 2 0 0 1 - 0 8 8 8 5 0]

出 願 人 関西ティー・エル・オー株式会社
Applicant(s):

2 0 0 3 年 8 月 5 日

特許庁長官
Commissioner,
Japan Patent Office

今 井 康



出証番号 出証特 2 0 0 3 - 3 0 6 2 7 5 9

【書類名】 特許願

【整理番号】 22039

【提出日】 平成13年 3月26日

【あて先】 特許庁長官殿

【国際特許分類】 G06F 12/02 510

【発明の名称】 関数実行方法、関数実行装置、コンピュータプログラム
、及び記録媒体

【請求項の数】 9

【発明者】

【住所又は居所】 東京都新宿区新宿 2 丁目 4 番 3 号 フォーシーズンビル
1 0 階 株式会社数理システム内

【氏名】 山本 晃成

【発明者】

【住所又は居所】 京都府京都市左京区吉田本町 京都大学内

【氏名】 湯浅 太一

【特許出願人】

【識別番号】 899000046

【氏名又は名称】 関西ティー・エル・オー株式会社

【代理人】

【識別番号】 100078868

【弁理士】

【氏名又は名称】 河野 登夫

【電話番号】 06(6944)4141

【手数料の表示】

【予納台帳番号】 001889

【納付金額】 21,000円

【提出物件の目録】

【物件名】 明細書 1

【物件名】 図面 1

【物件名】 要約書 1

【包括委任状番号】 9908225

【プルーフの要否】 要

【書類名】 明細書

【発明の名称】 関数実行方法、関数実行装置、コンピュータプログラム、及び記録媒体

【特許請求の範囲】

【請求項 1】 メモリ中のスタック領域に、他の関数を呼び出す処理を含む呼出関数から呼び出される被呼出関数の形式に基づく関数記録領域を積み上げ、積み上げた関数記録領域を利用して被呼出関数を呼び出し、呼び出された被呼出関数の実行後、積み上げた関数記録領域を破棄する関数実行方法において、

関数記録領域で実行される呼出関数を解析し、

該解析により呼出関数が所定の条件を満足すると判断した場合に、呼出関数を実行する前記関数記録領域を、被呼出関数を呼び出す領域として利用する

ことを特徴とする関数実行方法。

【請求項 2】 前記所定の条件は、被呼出関数の実行結果が呼出関数の実行結果となることであることを特徴とする請求項 1 に記載の関数実行方法。

【請求項 3】 前記呼出関数及び被呼出関数が異なるときに、被呼出関数の形式に基づいて、前記関数記録領域を変更することを特徴とする請求項 1 又は請求項 2 に記載の関数実行方法。

【請求項 4】 メモリ中のスタック領域に、他の関数を呼び出す処理を含む呼出関数から呼び出された被呼出関数の形式に基づく関数記録領域を積み上げ、積み上げた関数記録領域を利用して被呼出関数を呼び出し、呼び出された被呼出関数の実行後、積み上げた関数記録領域を破棄する関数実行装置において、

関数記録領域で実行される呼出関数を解析する手段と、

該解析により呼出関数が所定の条件を満足すると判断した場合に、呼出関数を実行する前記関数記録領域を、被呼出関数を呼び出す領域として利用する手段とを備えることを特徴とする関数実行装置。

【請求項 5】 コンピュータに、メモリ中のスタック領域に、他の関数を呼び出す処理を含む呼出関数から呼び出される被呼出関数の形式に基づく関数記録領域を積み上げさせ、積み上げた関数記録領域を利用して被呼出関数を呼び出させ、呼び出された被呼出関数の実行後、積み上げた関数記録領域を破棄させるコ

ンピュータプログラムにおいて、

コンピュータに、関数記録領域で実行される呼出関数を解析させる手順と、

コンピュータに、解析により呼出関数が所定の条件を満足すると判断した場合に、呼出関数を実行する前記関数記録領域を、被呼出関数を呼び出す領域として利用させる手順と

を含むことを特徴とするコンピュータプログラム。

【請求項 6】 前記所定の条件は、被呼出関数の実行結果が呼出関数の実行結果となることであることを特徴とする請求項 5 に記載のコンピュータプログラム。

【請求項 7】 前記呼出関数及び被呼出関数が異なるときに、被呼出関数の形式に基づいて、前記関数記録領域を変更することを特徴とする請求項 5 又は請求項 6 に記載のコンピュータプログラム。

【請求項 8】 コンピュータに、メモリ中のスタック領域に、他の関数を呼び出す処理を含む呼出関数から呼び出される被呼出関数の形式に基づく関数記録領域を積み上げさせ、積み上げた関数記録領域を利用して被呼出関数を呼び出させ、呼び出された被呼出関数の実行後、積み上げた関数記録領域を破棄させるコンピュータプログラムにおいて、

コンピュータに、関数記録領域で実行される第 1 呼出関数を解析させる手順と、

コンピュータに、解析により第 1 呼出関数が所定の条件を満足すると判断した場合に、呼出関数を実行する前記関数記録領域を、被呼出関数を呼び出す領域として利用させる処理を含む第 1 と異なる第 2 呼出関数を、第 1 呼出関数の代替関数として実行させる手順と

を含むことを特徴とするコンピュータプログラム。

【請求項 9】 コンピュータに、メモリ中のスタック領域に、他の関数を呼び出す処理を含む呼出関数から呼び出される被呼出関数の形式に基づく関数記録領域を積み上げさせ、積み上げた関数記録領域を利用して被呼出関数を呼び出させ、呼び出された被呼出関数の実行後、積み上げた関数記録領域を破棄させるコンピュータプログラムを記録してある、コンピュータでの読み取りが可能な記録

媒体において、

コンピュータに、関数記録領域で実行される呼出関数を解析させる手順と、

コンピュータに、解析により呼出関数が所定の条件を満足すると判断した場合に、呼出関数を実行する前記関数記録領域を、被呼出関数を呼び出す領域として利用させる手順と、

を含むコンピュータプログラムを記録してあることを特徴とするコンピュータでの読み取りが可能な記録媒体。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】

メモリ中のスタック領域に、他の関数を呼び出す処理を含む呼出関数から呼び出される被呼出関数の形式に基づく関数記録領域を積み上げ、積み上げた関数記録領域を利用して被呼出関数を呼び出し、呼び出された被呼出関数の実行後、当該関数領域を破棄する関数実行方法、その方法を適用した関数実行装置、その装置を実現するためのコンピュータプログラム、及びそのコンピュータプログラムを記録した記録媒体に関し、特にJVM(Java Virtual Machine)にてJava言語等のオブジェクト指向の言語にて記述された関数を実行する関数実行方法、関数実行装置、コンピュータプログラム、及び記録媒体に関する。

【0002】

【従来の技術】

Java言語等のオブジェクト指向の言語にて記述された複数の命令からなる関数を複数含むプログラムを実行する関数実行方法が近年様々な用途で利用されており、またこのようなJava言語は一般的にJVMのバイトコードにコンパイルされてからJVMのメモリ中の領域にて実行される。

【0003】

図7はプログラムの実行に必要な関数を記録する関数記録領域を示す概念図である。

プログラムは、一般に関数(メソッド)の呼び出しが入れ子状態になって実行される。即ち図7(a), (b), (c)に示すようにプログラムの処理に必要な

な関数は、実行中の呼出関数から実行すべき被呼出関数を呼び出して、メモリ中のスタック領域に確保された実行中の呼出関数の関数記録領域（フレーム）に、被呼出関数の形式に基づく新たな関数記録領域を積み上げる形で確保し、確保された関数記録領域を利用して呼び出された被呼出関数を実行し、被呼出関数の実行完了後、積み上げられている不要になった関数記録領域を破棄する。

【0004】

最上位に積み上げられた実行中の関数を記録している関数記録領域は、トップフレームと呼ばれ、図7（a）では関数Fが実行中であり、関数Fの関数記録領域がトップフレームとなる。

この関数Fを呼出関数として、被呼出関数である関数Gが呼び出されると、図7（b）に示すように、関数Gを実行するための関数記録領域がスタック領域に積み上げられてトップフレームとなり、関数Gの実行完了後、関数Gの実行結果を関数Fに渡し、当該関数記録領域は破棄されて、呼び出し前の図7（a）の状態となる。

【0005】

なお図7（b）において、関数Gを呼出関数として、更に他の被呼出関数である関数Hが呼び出されたときは、図7（c）に示すように、関数Hを実行するための関数記録領域がスタック領域に積み上げられてトップフレームとなり、関数Hの実行完了後、関数Hの実行結果を関数Gに渡し、当該関数記録領域は破棄されて、呼び出し前の図7（b）の状態となる。

【0006】

このとき呼出関数である関数Gにより呼び出された被呼出関数である関数Hの実行結果を関数Gの実行結果として関数Fに渡す場合、関数Hの呼び出しを末尾呼び出しと呼び、末尾呼び出しでは関数Hの実行完了後、関数H及び関数Gの関数記録領域は連続して破棄されることになる。

【0007】

【発明が解決しようとする課題】

しかしながらスタック領域に数多くの関数記録領域を積み上げていくことにより、スタック領域をオーバーフローするという状況を引き起こし、場合によって

はプログラムの実行に支障を来すことがあるという問題がある。

【0008】

また関数記録領域の積み上げ及び破棄に要する処理負荷が、全体の実行速度の低下を招くという問題がある。

【0009】

本発明は斯かる事情に鑑みてなされたものであり、呼出関数による呼び出しが末尾呼び出しであると判断した場合に、新たな関数記録領域を積み上げることなく、呼出関数が記録されている関数記録領域を、被呼出関数を記録する関数記録領域として利用することで、関数記録領域がスタック領域をオーバーフローしてプログラムの実行に支障を来す可能性を低減し、また積み上げ及び破棄に要する処理負荷を低減し、全体の実行速度を向上させる関数実行方法、その方法を適用した関数実行装置、その装置を実現するためのコンピュータプログラム、及びそのコンピュータプログラムを記録した記録媒体の提供を主たる目的とする。

【0010】

さらに呼出関数及び被呼出関数が同じ場合には、確保されている関数記録領域をそのまま利用し、呼出関数及び被呼出関数が異なる場合には、確保されている関数記録領域を被呼出関数の形式に基づいて変更することにより、全体の処理負荷の軽減及び関数記録領域の適正化を行うことが可能な関数実行方法等の提供を他の目的とする。

【0011】

【課題を解決するための手段】

第1発明に係る関数実行方法は、メモリ中のスタック領域に、他の関数を呼び出す処理を含む呼出関数から呼び出される被呼出関数の形式に基づく関数記録領域を積み上げ、積み上げた関数記録領域を利用して被呼出関数を呼び出し、呼び出された被呼出関数の実行後、積み上げた関数記録領域を破棄する関数実行方法において、関数記録領域で実行される呼出関数を解析し、該解析により呼出関数が所定の条件を満足すると判断した場合に、呼出関数を実行する前記関数記録領域を、被呼出関数を呼び出す領域として利用することを特徴とする。

【0012】

第1発明に係る関数実行方法では、呼出関数が所定の条件を満足した場合に、呼出関数の関数記録領域を被呼出関数の記録領域として再利用して、スタック領域に積み上げられる関数記録領域の数を低減することで、関数記録領域がスタック領域をオーバーフローしてプログラムの実行に支障を来す可能性を低減し、また関数記録領域の積み上げ及び破棄に要する処理負荷を低減して、全体の実行速度を向上させることが可能である。

【0013】

第2発明に係る関数実行方法は、第1発明において、前記所定の条件は、被呼出関数の実行結果が呼出関数の実行結果となることであることを特徴とする。

【0014】

第2発明に係る関数実行方法では、呼出関数による呼び出しが末尾呼び出し、即ち被呼出関数の実行結果が呼出関数の実行結果となり、被呼出関数の実行完了後、被呼出関数の関数記録領域に続いて、呼出関数の関数記録領域も破棄される呼び出し、例えばJava言語の場合では被呼出関数を呼び出す命令の直後に任意個のプログラムカウンタ以外を変化させない命令を挟んで復帰命令があり、被呼出関数の戻り値の型と復帰命令の型とが一致し、被呼出関数を呼び出す命令から復帰命令の間に例外ハンドラが設定されていないという基準を満足することを条件とすることにより、呼出関数の関数記録領域を被呼出関数の関数記録領域として問題を発生することなく再利用することが可能である。

【0015】

第3発明に係る関数実行方法は、第1発明又は第2発明において、前記呼出関数及び被呼出関数が異なるときに、被呼出関数の形式に基づいて、前記関数記録領域を変更することを特徴とする。

【0016】

第3発明に係る関数実行方法では、呼出関数及び被呼出関数が異なるときに、呼出関数のために確保されている関数記録領域を被呼出関数の形式に基づいて変更することにより、被呼出関数を再利用した関数記録領域にて問題なく実行することが可能であり、また呼出関数及び被呼出関数が同じ関数であるときには、関数記録領域の形式を変更することなく、そのまま再利用するので形式の変更に要

する処理を無くし、全体としての処理速度を向上させることが可能である。

【0017】

第4発明に係る関数実行装置は、メモリ中のスタック領域に、他の関数を呼び出す処理を含む呼出関数から呼び出された被呼出関数の形式に基づく関数記録領域を積み上げ、積み上げた関数記録領域を利用して被呼出関数を呼び出し、呼び出された被呼出関数の実行後、積み上げた関数記録領域を破棄する関数実行装置において、関数記録領域で実行される呼出関数を解析する手段と、該解析により呼出関数が所定の条件を満足すると判断した場合に、呼出関数を実行する前記関数記録領域を、被呼出関数を呼び出す領域として利用する手段とを備えることを特徴とする。

【0018】

第4発明に係る関数実行装置では、呼出関数による呼び出しが、被呼出関数の実行結果を呼出関数の実行結果とする末尾呼び出しである等の所定の条件を満足すると判断した場合に、呼出関数の関数記録領域を被呼出関数の記録領域として再利用し、スタック領域に積み上げられる関数記録領域の数を低減することで、関数記録領域がスタック領域をオーバーフローしてプログラムの実行に支障を来す可能性を低減し、また関数記録領域の積み上げ及び破棄に要する処理負荷を低減して、全体の実行速度を向上させることが可能である。

【0019】

第5発明に係るコンピュータプログラムは、コンピュータに、メモリ中のスタック領域に、他の関数を呼び出す処理を含む呼出関数から呼び出される被呼出関数の形式に基づく関数記録領域を積み上げさせ、積み上げた関数記録領域を利用して被呼出関数を呼び出させ、呼び出された被呼出関数の実行後、積み上げた関数記録領域を破棄させるコンピュータプログラムにおいて、コンピュータに、関数記録領域で実行される呼出関数を解析させる手順と、コンピュータに、解析により呼出関数が所定の条件を満足すると判断した場合に、呼出関数を実行する前記関数記録領域を、被呼出関数を呼び出す領域として利用させる手順とを含むことを特徴とする。

【0020】

第5発明に係るコンピュータプログラムでは、携帯電話及びパーソナルコンピュータ等の処理装置を用いたJVMにて実行することで、JVMが関数実行装置として動作することにより、呼出関数が所定の条件を満足した場合に、呼出関数の関数記録領域を被呼出関数の記録領域として再利用し、スタック領域に積み上げられる関数記録領域の数を低減することで、関数記録領域がスタック領域をオーバーフローしてプログラムの実行に支障を来す可能性を低減し、また関数記録領域の積み上げ及び破棄に要する処理負荷を低減して、全体の実行速度を向上させることが可能である。

【0021】

第6発明に係るコンピュータプログラムは、第5発明において、前記所定の条件は、被呼出関数の実行結果が呼出関数の実行結果となることであることを特徴とする。

【0022】

第6発明に係るコンピュータプログラムでは、呼出関数による呼び出しが末尾呼び出し、即ち被呼出関数の実行結果が呼出関数の実行結果となり、被呼出関数の実行完了後、被呼出関数の関数記録領域に続いて、呼出関数の関数記録領域も破棄される呼び出しであることを条件とすることにより、呼出関数の関数記録領域を被呼出関数の関数記録領域として問題を発生することなく再利用することが可能である。

【0023】

第7発明に係るコンピュータプログラムは、第5発明又は第6発明において、前記呼出関数及び被呼出関数が異なるときに、被呼出関数の形式に基づいて、前記関数記録領域を変更することを特徴とする。

【0024】

第7発明に係るコンピュータプログラムでは、呼出関数及び被呼出関数が異なる関数であるときに、呼出関数のために確保されている関数記録領域を被呼出関数の形式に基づいて変更することにより、被呼出関数を再利用した関数記録領域にて問題なく実行することが可能であり、また呼出関数及び被呼出関数が同じ関数であるときには、関数記録領域の形式を変更することなく、そのまま再利用す

るので形式の変更に要する処理を無くし、全体としての処理速度を向上させることが可能である。

【0025】

第8発明に係るコンピュータプログラムは、コンピュータに、メモリ中のスタック領域に、他の関数を呼び出す処理を含む呼出関数から呼び出される被呼出関数の形式に基づく関数記録領域を積み上げさせ、積み上げた関数記録領域を利用して被呼出関数を呼び出させ、呼び出された被呼出関数の実行後、積み上げた関数記録領域を破棄させるコンピュータプログラムにおいて、コンピュータに、関数記録領域で実行される第1呼出関数を解析させる手順と、コンピュータに、解析により第1呼出関数が所定の条件を満足すると判断した場合に、呼出関数を実行する前記関数記録領域を、被呼出関数を呼び出す領域として利用させる処理を含む第1と異なる第2呼出関数を、第1呼出関数の代替関数として実行させる手順とを含むことを特徴とする。

【0026】

第8発明に係るコンピュータプログラムでは、携帯電話及びパーソナルコンピュータ等の処理装置を用いたJVMにて実行することで、JVMが関数実行装置として動作することにより、第1呼出関数による呼び出しが、被呼出関数の実行結果を呼出関数の実行結果とする末尾呼び出しである等の所定の条件を満足すると判断した場合に、利用した関数記録領域を被呼出関数の記録領域として再利用する処理を含む新たに用意された関数である第2呼出関数を、第1呼出関数の代替関数として実行し、これによりスタック領域に積み上げられる関数記録領域の数を低減して、関数記録領域がスタック領域をオーバーフローしてプログラムの実行に支障を来す可能性を低減し、また関数記録領域の積み上げ及び破棄に要する処理負荷を低減して、全体の実行速度を向上させることが可能である。

【0027】

第9発明に係るコンピュータでの読み取りが可能な記録媒体は、コンピュータに、メモリ中のスタック領域に、他の関数を呼び出す処理を含む呼出関数から呼び出される被呼出関数の形式に基づく関数記録領域を積み上げさせ、積み上げた関数記録領域を利用して被呼出関数を呼び出させ、呼び出された被呼出関数の実

行後、積み上げた関数記録領域を破棄させるコンピュータプログラムを記録してある、コンピュータでの読み取りが可能な記録媒体において、コンピュータに、関数記録領域で実行される呼出関数を解析させる手順と、コンピュータに、解析により呼出関数が所定の条件を満足すると判断した場合に、呼出関数を実行する前記関数記録領域を、被呼出関数を呼び出す領域として利用させる手順と、を含むコンピュータプログラムを記録してあることを特徴とする。

【0028】

第9発明に係るコンピュータでの読み取りが可能な記録媒体では、記録されているコンピュータプログラムを携帯電話及びパーソナルコンピュータ等の処理装置を用いたJVMにて実行することで、JVMが関数実行装置として動作することにより、呼出関数が所定の条件を満足した場合に、呼出関数の関数記録領域を被呼出関数の記録領域として再利用し、スタック領域に積み上げられる関数記録領域の数を低減することで、関数記録領域がスタック領域をオーバーフローしてプログラムの実行に支障を来す可能性を低減し、また関数記録領域の積み上げ及び破棄に要する処理負荷を低減して、全体の実行速度を向上させることが可能である。

【0029】

【発明の実施の形態】

以下、本発明をその実施の形態を示す図面に基づいて詳述する。

図1は本発明の関数実行装置を示すブロック図である。

図中10は携帯電話及びパーソナルコンピュータ等の処理装置を用いた本発明の関数実行装置であり、関数実行装置10は、本発明の関数実行装置用のコンピュータプログラム及びデータ等の情報を記録したCD-ROM及びメモ리카ード等の記録媒体RECからコンピュータプログラム及びデータ等の情報を読み取る補助記憶手段12、補助記憶手段12により読み取られたコンピュータプログラム及びデータ等の情報を記録するハードディスク等の記録手段13、並びに各種情報を一時的に記録するメモリ手段14を備えている。

そして記録手段13からコンピュータプログラム及びデータ等の情報を読み取り、メモリ手段14に記録してCPU11により実行することで、処理装置（コ

ンピュータ)は本発明の関数実行装置として動作する。

【0030】

また関数実行装置10は、モデム、T A(Terminal Adapter)、及びアンテナ等の通信手段15を備えており、通信手段15によりインターネット等の通信ネットワークNWに接続して、通信ネットワークNWに接続するウェブサーバコンピュータ等の記録装置20が備える記録媒体21に記録された本発明のコンピュータプログラム及びデータ等の情報を取り込み、実行するようにしてもよい。

【0031】

次に本発明の関数実行方法の処理内容を説明する。

本発明の関数実行方法は、J a v a言語等の言語にて記述され、J V Mのバイトコードにコンパイルされた複数の命令からなる関数を複数含むプログラムを実行する方法に適用されるものであり、メモリ手段14中のスタック領域に、実行すべき関数についての引数の個数及びローカル変数領域の大きさ等の形式に基づく関数記録領域を積み上げ、積み上げた関数記録領域に、関数を呼び出して実行する関数実行方法を基本とする。

【0032】

図2は本発明の関数実行方法を概念的に示す説明図である。

図2(a)はスタック関数領域に確保された関数記録領域にて関数Fが実行されている状態を示す。

そして関数記録領域にて実行される関数Fを呼出関数とし、関数Fに呼び出される被呼出関数である関数Gを呼び出す場合、図2(b)に示すように関数Fの関数記録領域に、関数Gを呼び出すための関数記録領域を新たに積み上げ、積み上げた関数記録領域を利用して関数Gを呼び出し実行する。

【0033】

さらに関数Gを呼出関数とし、関数Gに呼び出される被呼出関数である関数Hを呼び出す場合で、関数Hの実行結果が関数Gの実行結果となる末尾呼出であるときに、図2(c)に示すように関数Hのための関数記録領域を新たに積み上げることなく、関数Gが記録されている関数記録領域を、関数Hを呼び出すための関数記録領域とする。

そして関数Hの実行結果は関数Fに渡され、関数Hを実行した関数記録領域は破棄される。

【0034】

次に本発明の関数実行方法における解析処理を図3に示すフローチャートを用いて説明する。

JVM等のバイトコードによる関数を実行するにあたり、先ず新たに関数記録領域を確保し、そして呼び出すべき関数が、`invokestatic`、`invokevirtual`、`invokespecial`、及び`invokeinterface`等の被呼出関数を呼び出す命令を含む呼出関数である場合に、当該呼出関数を解析し(S101)、解析により、呼出関数が、被呼出関数の実行結果が呼出関数の実行結果となるという条件を満足する末尾呼出関数であると判断したときに(S102:Y)、更に呼出関数及び被呼出関数が同じであるか否かを判別する(S103)。

なおステップS102において末尾呼出関数であると判断する基準のJava言語における具体例としては、被呼出関数を呼び出す命令の直後に任意個のプログラムカウンタ以外を変化させない`nop`及び`goto`等の命令を挟んで復帰命令があり、被呼出関数の戻り値の型と、`ireturn`、`lreturn`、`freturn`、`dreturn`、`areturn`、及び`return`等の復帰命令の型とが一致し、被呼出関数を呼び出す命令から復帰命令の間に例外ハンドラが設定されていない等の基準がある。

【0035】

ステップS103において、呼出関数及び被呼出関数が異なると判別したとき(S103:N)、当該呼出関数(第1呼出関数)を、被呼出関数を呼び出す命令を予め用意している代替命令に置換した再帰代替関数(第2呼出関数)に置換する(S104)。

呼出関数及び被呼出関数が同じであると判別したとき(S103:Y)当該呼出関数(第1呼出関数)を、呼出関数と同じである被呼出関数を呼び出す命令を予め用意している代替命令に置換した自己再帰代替関数(第2呼出関数)に置換する(S105)。

なお置換すべき命令が複数個含まれる場合、該当する全ての命令を置換した関数に置換される。

ステップ S102 において、末尾呼出関数でないと判断したとき (S102 : N)、ステップ S103 ~ S105 の処理は行われない。

【0036】

なお呼出関数の代替関数として用いられるステップ S104 に示す再帰代替関数及びステップ S105 に示す自己再帰代替関数として、以下に示す新たな命令を含む関数を用意しておく。

即ち、

`invokestatic、`
`invokevirtual、`
`invokespecial、`
及び `invokeinterface`

等の呼出命令を含む呼出関数の再帰代替関数として、

`tailinvokestatic、`
`tailinvokevirtual、`
`tailinvokespecial、`
及び `tailinvokeinterface`

等の代替命令を用意し、用意した代替命令を含む再帰代替関数を用いる。

また自己再帰代替関数として、

`selftailinvokestatic、`
`selftailinvokevirtual、`
`selftailinvokespecial、`
及び `selftailinvokeinterface`

等の代替命令を用意し、用意した代替命令を含む自己再帰代替関数を用いる。

これらの命令を含む関数は、解析処理では置換されるだけであり、実際に実行されるのは、置換された関数を実行する実行処理においてであるので、以下の実行処理の説明にてその機能を説明する。

【0037】

次に本発明の関数実行方法における実行処理を図4及び図5に示すフローチャートを用いて説明する。

解析処理により必要に応じて置換がなされた関数を実行する場合、先ず実行すべき呼出関数が、代替関数である再帰代替関数或いは自己再帰代替関数、又は代替関数で無いかを判別し（S201）、再帰代替関数であると判断した場合（S201：1）、再帰代替関数（第2呼出関数）の形式に基づく関数記録領域を積み上げ（S202）、積み上げた関数記録領域を利用して再帰代替関数を呼び出し（S203）、ステップS202にて積み上げた関数記録領域にて再帰代替関数を実行する（S204）

【0038】

そして代替関数に含まれる `tail invoke static` 等の代替命令による処理により、呼出関数（第1呼出関数）として実行している再帰代替関数（第2呼出関数）から被呼出関数を呼び出す場合に、新たに関数記録領域を積み上げることなく、再帰代替関数を実行した関数記録領域を、被呼出関数の形式に基づいて変更し（S205）、形式を変更した関数記録領域を、被呼出関数を呼び出す領域として利用して（S206）、被呼出関数を呼び出し（S207）、呼び出した被呼出関数を実行し（S208）、被呼出関数の実行後、被呼出関数の実行に利用した関数記録領域を破棄する（S209）。

【0039】

ステップS201において、自己再帰代替関数であると判断した場合（S201：2）、自己再帰代替関数（第2呼出関数）の形式に基づく関数記録領域を積み上げ（S210）、積み上げた関数記録領域を利用して自己再帰代替関数を呼び出し（S211）、ステップ210にて積み上げた関数記録領域にて自己再帰代替関数を実行する（S212）。

【0040】

そして自己再帰代替関数に含まれる `self tail invoke static` 等の代替命令による処理により、呼出関数（第1呼出関数）として実行している自己再帰代替関数（第2呼出関数）から被呼出関数を呼び出す場合に、新たに関数記録領域を積み上げることなく、自己再帰代替関数を実行した関数記録領域

を、被呼出関数を呼び出す領域として利用して（S 2 1 3）、被呼出関数を呼び出し（S 2 1 4）、呼び出した被呼出関数を実行し（S 2 1 5）、被呼出関数の実行後、被呼出関数の実行に利用した関数記録領域を破棄する（S 2 1 6）。

【0041】

ステップ S 2 0 1 において、代替関数でないと判断した場合（S 2 0 1 : 3）、呼出関数の形式に基づく関数記録領域を積み上げ（S 2 1 7）、積み上げた関数記録領域を利用して呼出関数を呼び出し（S 2 1 8）、ステップ 2 1 7 にて積み上げられた関数記録領域にて呼出関数を実行する（S 2 1 9）。

【0042】

そして呼出関数から被呼出関数を呼び出す場合に、被呼出関数の形式に基づく新たな関数記録領域を積み上げ（S 2 2 0）、積み上げた関数記録領域を利用して被呼出関数を呼び出し（S 2 2 1）、ステップ S 2 2 0 にて積み上げた関数記録領域にて被呼出関数を実行し（S 2 2 2）、被呼出関数の実行後、被呼出関数の実行に利用した関数記録領域を破棄し（S 2 2 3）、更に呼出関数における被呼出関数実行後の処理を実行して、呼出関数の実行に利用した関数記録領域を破棄する（S 2 2 4）。

【0043】

次に本発明の関数実行方法及び従来の関数実行方法の処理速度を比較した結果を図 6 に示すグラフを用いて説明する。

図 6 では横軸に末尾呼出関数の呼び出し深さをとり、縦軸にマイクロ秒（ μs ）を単位とする処理時間をとって、その関係を示したものであり、×印は呼出関数を呼び出す都度、関数記録領域を積み上げる従来の関数実行方法による呼び出し深さと処理時間との関係を示し、□印は呼出関数が末尾呼出の場合に関数記録領域を再利用する本発明の関数実行方法による呼び出し深さと処理時間との関係を示し、○印は呼出関数と被呼出関数とが同じである自己末尾呼出関数を扱う場合の呼び出し深さと処理時間との関係を示している。

図 6 に示すように従来の関数実行方法と比較して、本発明の関数実行方法は処理時間が短く、特に自己末尾呼出関数ではその傾向が顕著である。

【0044】

また本発明の実施の形態としては、JIT(Just In Time Compiler)と呼ばれるJVMの実装技術を用いてもよく、JIT技術により、JVMのバイトコードを機械語に変換(コンパイル)して、CPU11にて実行することで、実行処理の速度を向上させることが可能である。

【0045】

そして前記実施の形態では、呼出関数が末尾呼出又は自己末尾呼出である場合に、代替関数を用いる形態を示したが、本発明はこれに限らず、関数実行時に都度再帰判定を行い、末尾再帰であると判断した場合に、末尾再帰処理を行うようにしてもよい。

【0046】

【発明の効果】

以上詳述した如く本発明に係る命令実行方法、命令実行装置、コンピュータプログラム、及び記録媒体では、Java言語等の言語にて記述され複数の関数を含むプログラムを、携帯電話及びパーソナルコンピュータ等の処理装置を用いたJVMにて実行する場合で、メモリ中のスタック領域に、他の関数を呼び出す処理を呼出関数による呼び出しが、被呼出関数の実行結果を呼出関数の実行結果とする末尾呼び出しである等の所定の条件を満足すると判断したときに、呼出関数の関数記録領域を被呼出関数の記録領域として再利用し、スタック領域に積み上げられる関数記録領域の数を低減することで、関数記録領域がスタック領域をオーバーフローしてプログラムの実行に支障を来す可能性を低減し、また関数記録領域の積み上げ及び破棄に要する処理負荷を低減して、全体の実行速度を向上させることが可能である等、優れた効果を奏する。

【0047】

また本発明では呼出関数及び被呼出関数が異なるときに、呼出関数のために確保されている関数記録領域を被呼出関数の形式に基づいて変更することにより、被呼出関数を再利用した関数記録領域にて問題なく実行することが可能であり、また呼出関数及び被呼出関数が同じ関数であるときには、関数記録領域の形式を変更することなく、そのまま再利用するので形式の変更に要する処理を無くし、全体としての処理速度を向上させることが可能である等、優れた効果を奏する。

【図面の簡単な説明】**【図 1】**

本発明の関数実行装置を示すブロック図である。

【図 2】

本発明の関数実行方法を概念的に示す説明図である。

【図 3】

本発明の関数実行方法における解析処理を示すフローチャートである。

【図 4】

本発明の関数実行方法における実行処理を示すフローチャートである。

【図 5】

本発明の関数実行方法における実行処理を示すフローチャートである。

【図 6】

本発明の関数実行方法及び従来の関数実行方法の処理速度を示すグラフである。

【図 7】

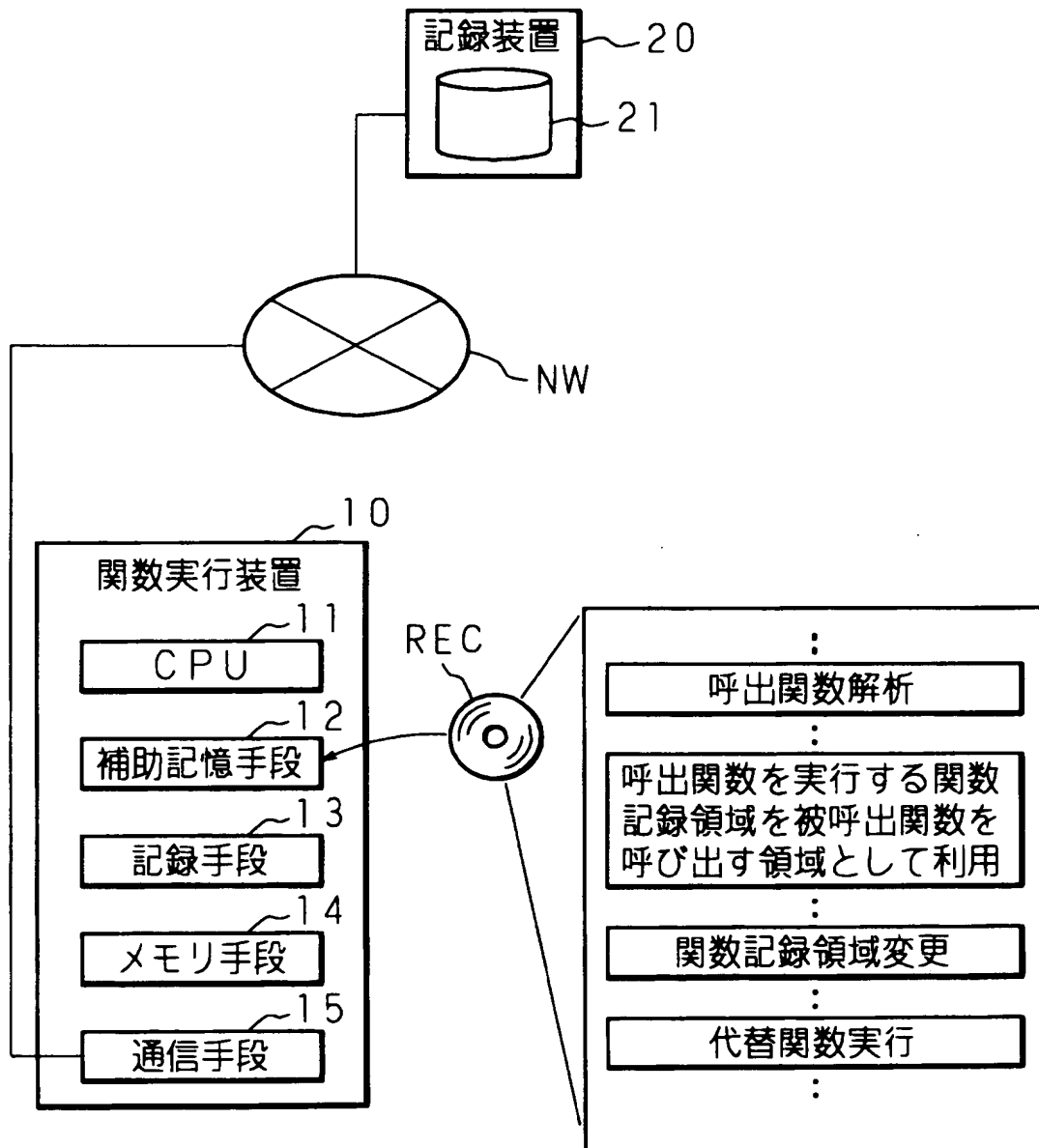
プログラムの実行に必要な関数を記録する関数記録領域を示す概念図である。

【符号の説明】

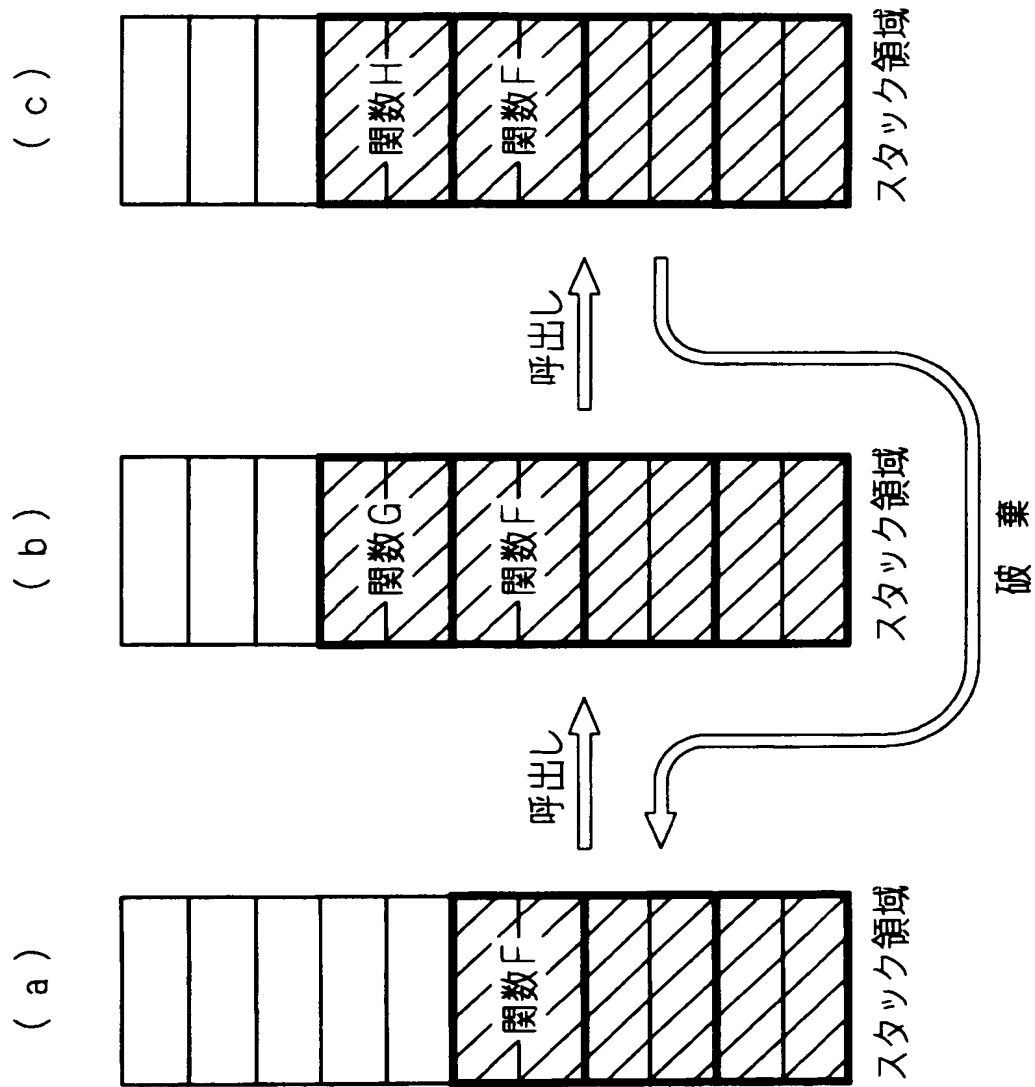
- 10 関数実行装置
- 11 CPU
- 12 補助記憶手段
- 13 記録手段
- 14 メモリ手段
- 15 通信手段
- 20 記録装置
- 21 記録媒体
- REC 記録媒体
- NW 通信ネットワーク

【書類名】 図面

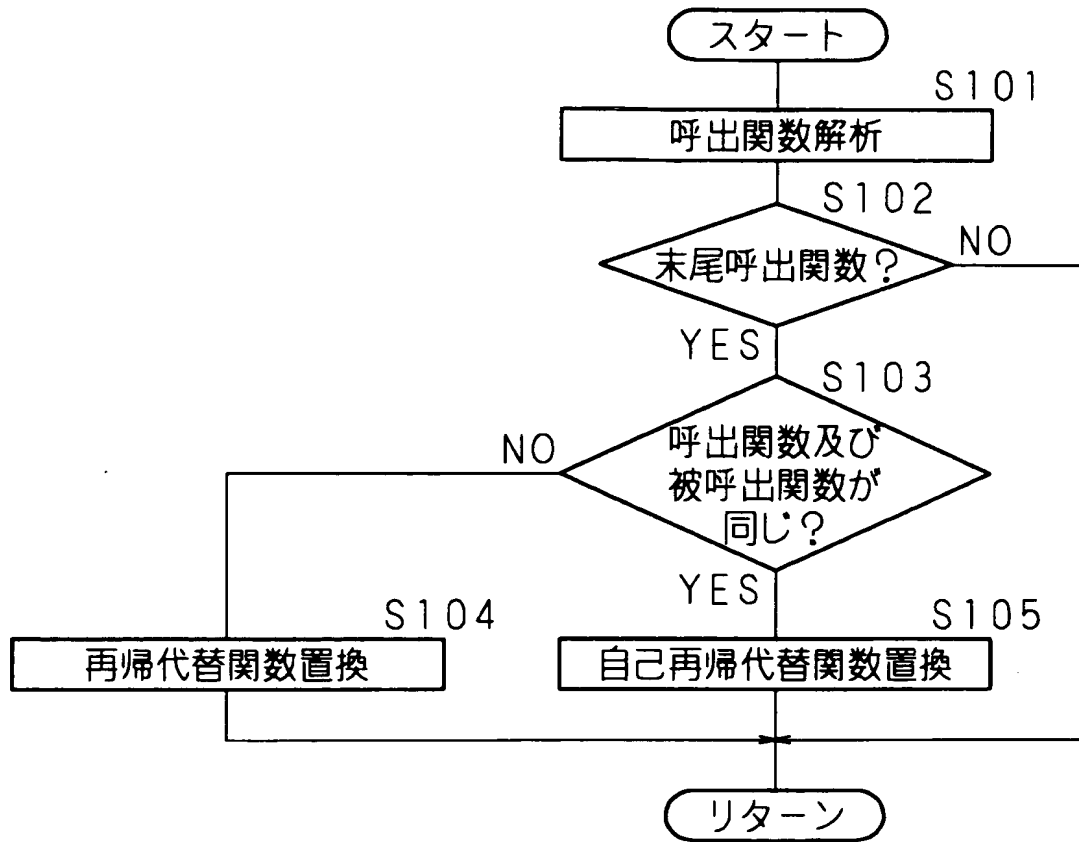
【図 1】



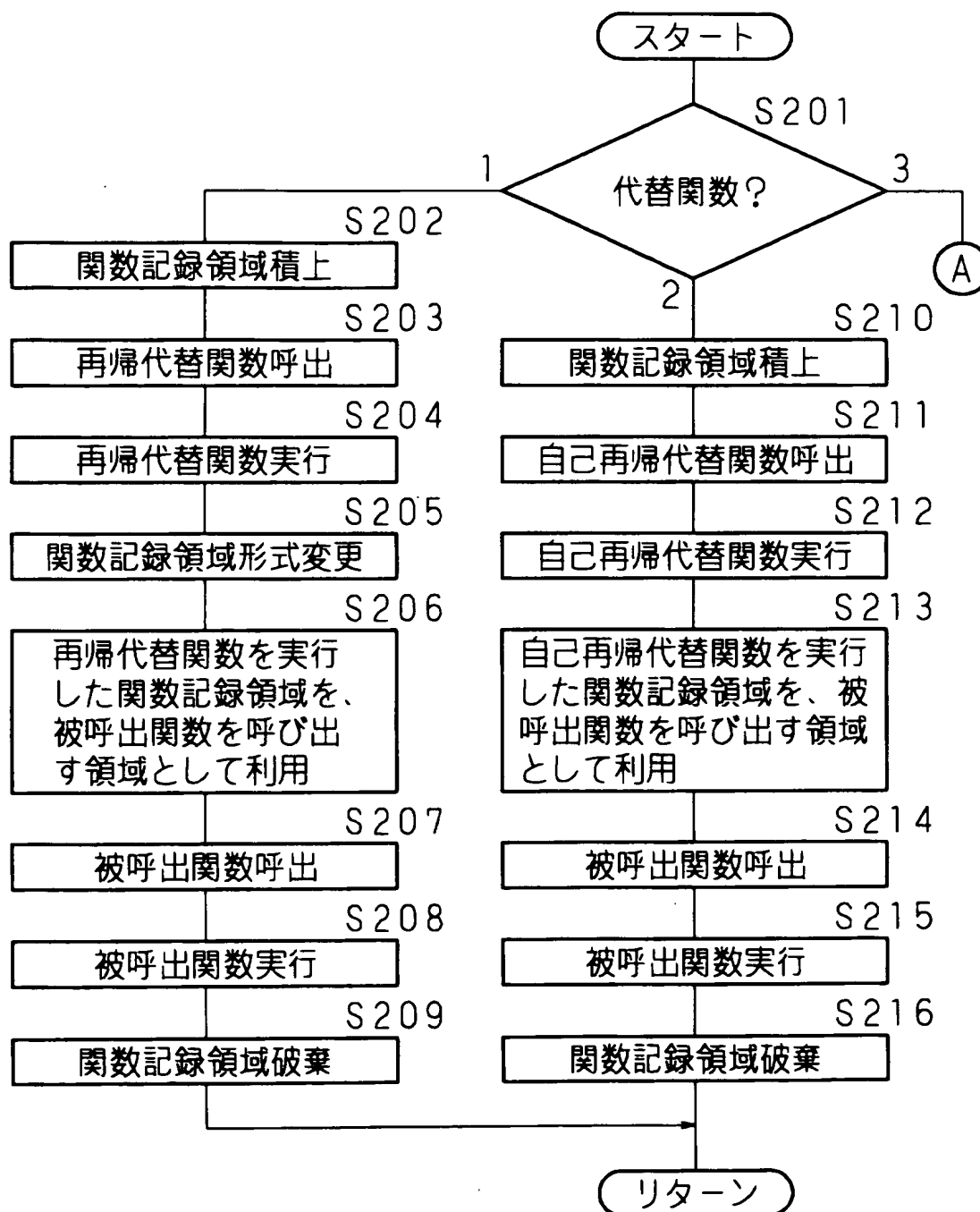
【図 2】



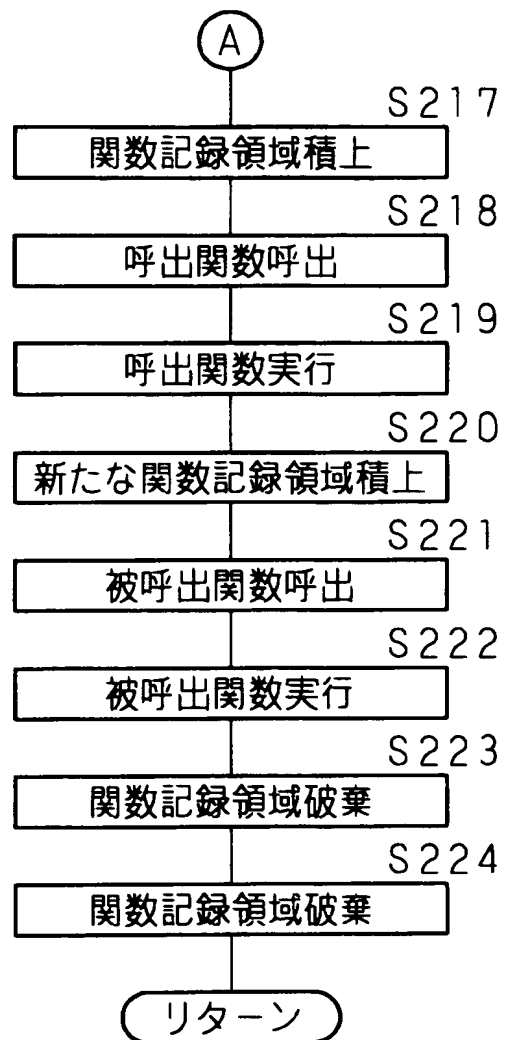
【図 3】



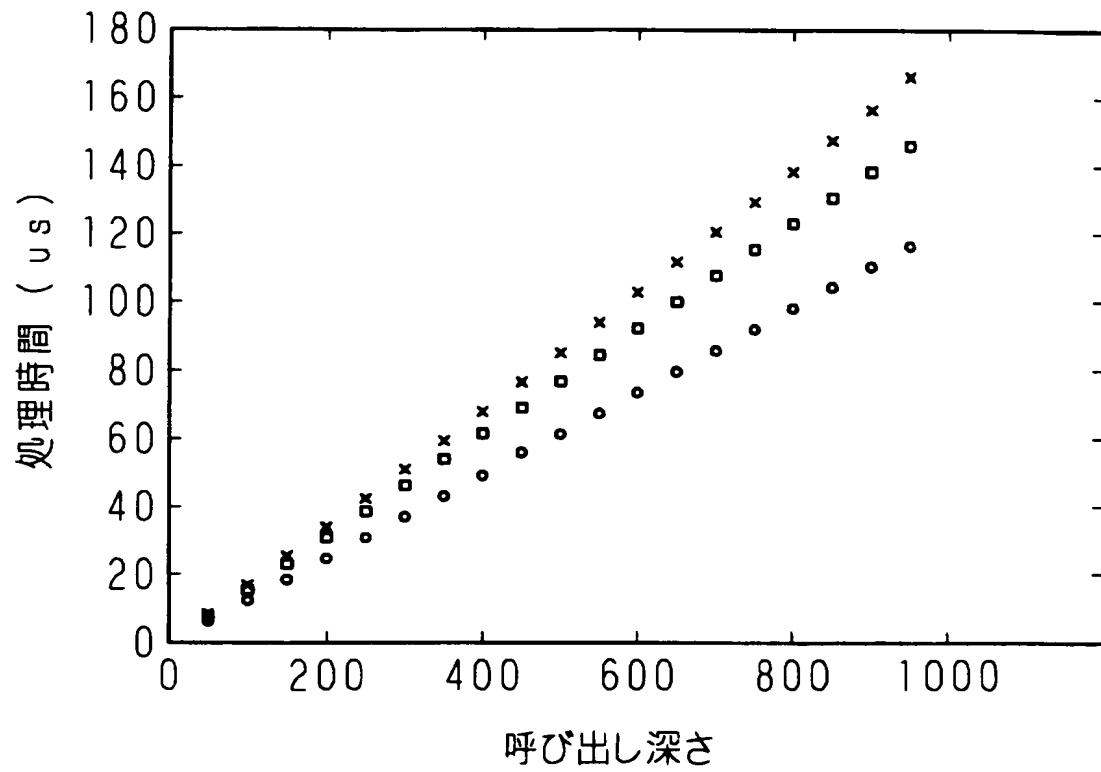
【図 4】



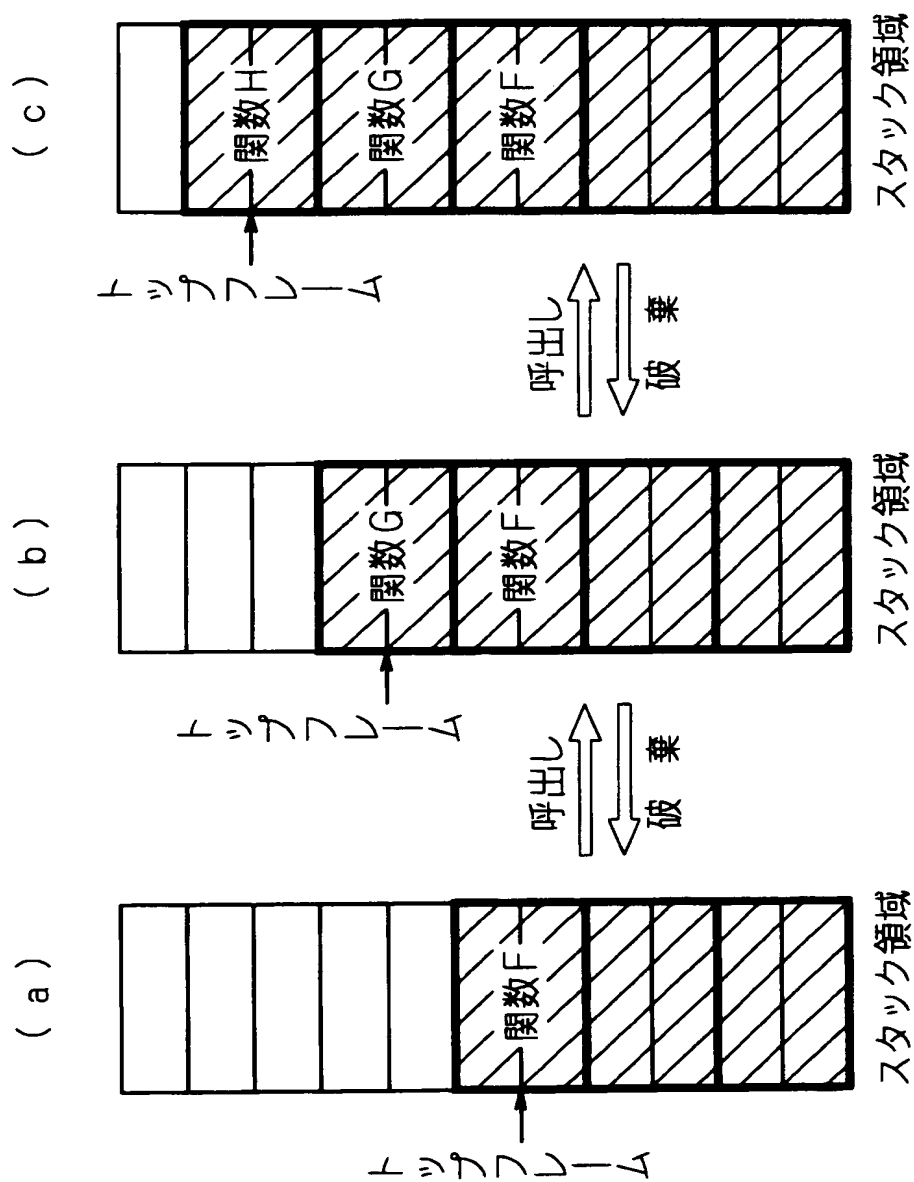
【図 5】



【図 6】



【図 7】



【書類名】 要約書

【要約】

【課題】 メモリ中のスタック領域に、他の関数を呼び出す処理を含む呼出関数から呼び出される被呼出関数の形式に基づく関数記録領域を積み上げて、被呼出関数を呼び出し、被呼出関数の実行後、積み上げた関数記録領域を破棄するプログラムの実行において、関数記録領域のオーバーフローを防止し、全体の実行速度を向上させる関数実行方法、関数実行装置、コンピュータプログラム、及び記録媒体を提供する。

【解決手段】 J V M等のバイトコードによる呼出関数を解析し、呼出関数が末尾呼出関数であると判断したときに、呼出関数を所定の代替関数に置換し、置換した代替関数を実行することで、代替関数である呼出関数から被呼出関数を呼び出す場合に、呼出関数の実行に利用した関数記録領域を利用して被呼出関数を呼び出し、呼び出した被呼出関数を実行する。

【選択図】 図 2

特願 2001-088850

出願人履歴情報

識別番号

[899000046]

1. 変更年月日 1999年 9月17日
[変更理由] 新規登録
住 所 京都府京都市下京区中堂寺栗田町1番地
氏 名 関西ティー・エル・オー株式会社
2. 変更年月日 2002年 8月 2日
[変更理由] 住所変更
住 所 京都府京都市下京区中堂寺栗田町93番地
氏 名 関西ティー・エル・オー株式会社